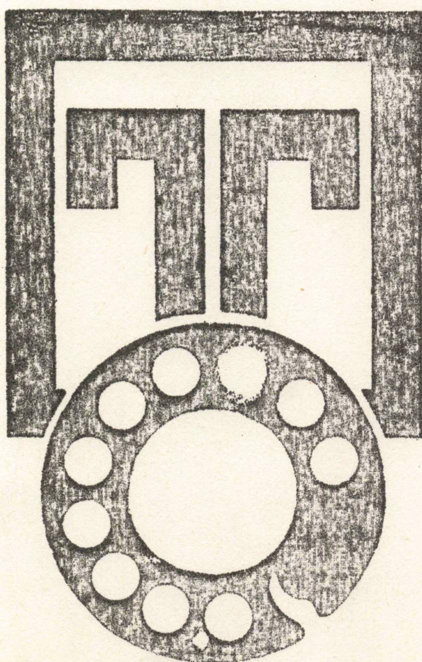


tietotehdas oy

Osituskäyttö

1.7.1970

BASIC-KIELEN ALKEET





BASIC-KURSSI

I Johdanto

Kieli on kehitetty 60-luvun alussa Yhdysvalloissa Darthmouth-Collegessa professorien Kemenyn ja Kurzin johdolla samoin kuin ensimmäinen ositus-käyttösysteemi. BASIC on lyhenne sanoista Beginners' Allpurpose Symbolic Instruction Code. Nimestään huolimatta se on tehtävän läheinen matemaattinen kieli, jota käyttävät muutkin kuin aloittelijat.

II Esimerkki I

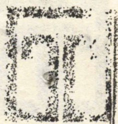
Alla on lyhyt BASIC-kielinen ohjelma

```
10 INPUT R
20 LET A = 3.1415*R↑2
30 PRINT A
40 END
```

Erikosta tekstissä on monelle, että

- ohjelmassa on vain suuria kirjaimia. Päätteessä, jolla ohjelmat kirjoitetaan, on myös ainoastaan suuret kirjaimet
- jokainen rivi alkaa numerolla. Näiden numeroiden avulla voidaan ohjelman käskyjen suoritusjärjestystä muuttaa ja niistä on paljon muutakin hyötyä.

Rivinumeroiden jälkeen välittömästi seuraavat englanninkieliset sanat, esimerkiksi INPUT, ovat BASIC-käskyjä, joiden merkitys sellaisina on täsmällisempi ja rajoitetumpi kuin normaalissa englanninkielessä. Rivinumeron jälkeen täytyy aina olla tällainen sana.



Sanojen, lukujen ja erilaisten merkkien väliin voidaan jättää tilaa kuinka runsaasti tahansa tai ne voidaan kirjoittaa yhteen. Ohjelman ensimmäinen rivi voitaisiin kirjoittaa esimerkiksi

```
10INPUTR
```

Ohjelman lauseet suoritetaan rivinumerojärjestyksessä, jollei toisin kahdella erityisellä käskyllä määrätä.

Lause 10 on kehoitus tietokoneelle pyytää käyttäjältä muuttujalle R arvo (INPUT R = SYÖTÄ MUUTTUJAN R ARVO SISÄÄN). Käyttäjä kirjoittaa haluamansa R:n arvon päätteellä koneen painettua paperille kysymysmerkin ja jäätyä odottamaan.

Lause 20 saa koneen laskemaan yhtäläisyysmerkin oikealla puolella olevan lausekkeen ja sijoittamaan saamansa tuloksen muuttujan A arvoksi.

Lauseen 30 kehoituksesta kone painaa muuttujan A arvon päätteessä olevalle listalle.

Lause 40 ilmoittaa koneelle, että suoritettavia lauseita ei ole enempää.

Esimerkissä on kaikki normaalin ohjelman osat:
- käsiteltävien tietojen sisäänsyöttöosa, INPUT-lause
- tietoa varsinaisesti käsittelevä osa, LET-lause
- haluttuun muotoon käsitellyn tiedon tulostusosa, PRINT-lause.

Kuitenkin ohjelma on melko hyödytön. Tuskin kukaan laskisi ympyrän alan sen avulla. Hyöty lisääntyy, jos ohjelmaa muunnetaan niin, että se laskee samalla ajokerralla useiden erisäteisten ympyröiden alan.

 tietotehdas oy

```
10 INPUT R
20 LET A = 3.1415*R↑2
30 PRINT A
35 GOTO 10
40 END
```

Tämän ohjelman suorittamista kone ei lopeta suoritettuaan lauseen 30, vaan uusi lause GOTO 10 saa koneen siirtymään riviltä 10 olevaa lausetta suorittamaan eli pyytämään käyttäjältä uuden säteen arvon. Kone suorittaa lauseita 10, 20, 30 ja 35, kunnes sen toiminta jollakin erikoistoimenpiteellä keskeytetään. Tällaista ohjelman osaa nimitetään silmukaksi eli loopiksi. Niitä käyttämällä saadaan kone tekemään paljon työtä vähällä vaivalla.

III Luvut

BASIC-kielessä on kolmenlaisia lukuja:

- kokonaislukuja, esimerkiksi 1, 123, -12
- desimaalilukuja, joita kutsutaan myös reaaliluvuiksi, esimerkiksi 2.3, -.2
- eksponenttilukuja, esimerkiksi -12.4E-3, mikä algebrassa merkittäisiin
  -12.4×10^{-3}

Desimaalipilkun asemesta käytetään pistettä. Pilkkua käytetään tässä systeemissä muihin tarkoituksiin.

Murtolukuja ei kielessä ole, kuten ei muitakaan edellä mainitsemattomia lukuja.

Luvut voidaan antaa koneelle korkeintaan 9 numeron tarkkuudella. Itseisarvoltaan suurin koneeseen mahtuva luku on ± 5.78960 E 76 ja lähinnä nollaa on luku ± 4.31809E -78.



IV Muuttujat

Ohjelmointikielissä ja matematiikassa käytetään muuttujia, jotka voivat saada erilaisia numeroarvoja. BASIC-kielen muuttujalle tietokone varaa muistiinsa paikan, missä muuttujan arvo säilytetään fyysisesti. Muuttujan nimessä on yksi kirjain tai yksi kirjain ja yksi numero. Siis sallitut muuttujien nimet ovat:

A AO AI A9
B BO BI B9
C CO CI C9
.
.
.
Z XO ZI ....

Muuttujille annetaan arvot LET-, READ- tai INPUT-lauseilla. Tämä arvo muuttuu vain, jos muuttujalle annetaan arvo uudella tällaisella lauseella. Muuttujalla on aina oltava arvo. Jos laskettavan lausekkeen kaikilla muuttujilla ei ole ennalta määrättyä arvoa, ei kone pysty lauseketta laskemaan.

V Operaattorit

I Aritmeettiset operaattorit
- yhteenlaskuoperaattori on tuttu "+"
- vähennyslaskuoperaattori on "-"
- kertolaskuoperaattori on tähti "*"
- jakolaskuoperaattori on kauttaviiva "/"
- potenssiinkorotusoperaattori on "↑"

Laskujärjestys on sama kuin matematiikassa. Järjestys voidaan muuttaa sulkumerkeillä "()", joita voi olla useita sisäkkäin.

2 Relaatio-operaattorit

Myöhemmin esitettävässä BASIC-käskyssä voidaan testata ehtoja. Tässä käskyssä

- merkki "=" tarkoittaa yhtäsuuruusrelaatiota
- merkki ">" suuremmuusrelaatiota
- merkki "<" pienemmyysrelaatiota
- merkki ">=" relaatiota, "suurempi tai yhtä suuri kuin"
- merkki "<=" relaatiota, "pienempi tai yhtä suuri kuin"
- merkki "<>" erisuuruusrelaatiota

VI Esimerkki 2

Tehtävänä on ratkaista yhtälöryhmä

$$ax + by = c$$
$$dx + ey = f$$

muuttamalla c:n ja f:n arvoilla. Parametrit a,b,d ja e pidetään vakioina, esimerkiksi a = 1, b = 2, d = 4, e = 2. Olkoot ensimmäisellä kerralla c = -7, f = 5, toisella kerralla c = 1, f = 3, kolmannella kerralla c = 4, f = -7.

Jos ae-bd ei ole 0,

$$X = \frac{ce - bf}{ae - bd} \qquad Y = \frac{af - cd}{ae - bd}$$

Jos ae-bd=0, ei yhtälöryhmällä ole ainakaan yksikäsitteistä ratkaisua.

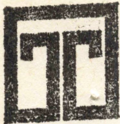
BASIC-kielellä koneelle kirjoitetut tehtävän suorittamisohjeet ovat tällaiset:

```
 10 READ A,B,D,E
 20 LET N =A*E  -B*D
 30 IF N = 0 THEN 90
 40 READ C,F
 50 LET X = (C*E  -B*F) /N
 60 LET Y = (A*F  -C*D) /N
 70 PRINT X, Y
 80 GOTO 40
 90 PRINT "YKSIKASITTEISIA RATKAISUA EI OLE"
100 DATA 1,2,4,2
110 DATA -7,5,1,3,4,-7
120 END
```

Lauseessa 10 on uusi BASIC-käsky, READ, joka liittyy kiinteästi riveillä 100 ja 110 oleviin käskyihin DATA. Kun ohjelma luetaan tietokoneeseen, käskyjen DATA jäljessä olevat luvut siirtyvät rivinumerojärjestyksessä muistiin yhdeksi käsiteltävien tietojen varastoksi. Ohjelmaa suoritettaessa DATA-lauseita ei käsitellä, vaan ne sivuutetaan. Jokainen käsky READ siirtää ensimmäisen käyttämättömän DATA-varastossa olevan luvun sen muuttujan arvoksi, jonka nimi on ensimmäisenä sanan READ jäljessä, toisen käyttämättömän luvun sen muuttujan arvoksi, jonka nimi on toisena sanan READ jäljessä, ja niin edelleen. Siis lauseessa kymmenen asetetaan muuttujan A arvoksi 1, muuttujan E arvoksi 2. Ensimmäiseksi käyttämättömäksi luvuksi jää -7.

Lause 20 on tuttu LET-lause. Siinä lasketaan tulojen ae ja bd erotus ja annetaan se muuttujan N arvoksi.

Lauseen 30 käsky, IF ehto THEN rivinumero, on toinen BASIC-kielen käsky, jolla käskyjen suoritusjärjestys voidaan muuttaa. Jos ehto, N=0, on tosi, suorittaa kone seuraavaksi lauseen 90. Jos N ei ole 0, suorittaa kone seuraavaksi lauseen, jonka numero on lähinnä suurempi, siis lauseen 40.

 tietotehdas oy

Lauseessa 40 sijoitetaan muuttujan C arvoksi ensimmäinen DATA-varaston käyttämätön luku, -7 ja muuttujan F arvoksi toinen DATA-varaston käyttämätön luku, 5.

Lauseessa 50 lasketaan tulojen ce ja bf erotus, jaetaan se aikaisemmin lasketulla tuntemattomien nimittäjällä ja sijoitetaan tulos muuttujan X arvoksi.

Lause 60 on samanlainen kuin 50.

Lauseen 70 kehoituksesta kone painaa päätteen listalle hiukan erilleen toisistaan muuttujien X ja Y arvot.

Lauseen 80 käsky GOTO 40 saa koneen suorittamaan käskyt 40,50,60,70,80 yhä uudelleen kunnes DATA-varastosta ei riitä uusia arvoja muuttujille C ja F. Tässä ohjelmassa on siis myös silmukka.

Lauseen 90 kone suorittaa vain, jos ae-bd on 0. Tällöin kone painaa listalle juuri ne merkit, jotka ovat käskyn PRINT jäljessä lainausmerkkien sisällä, siis

YKSIKASITTEISTA RATKAISUA EI OLE

Lause 120 on tieto koneelle, että ohjelmassa ei ole useampia lauseita.

VII Virheiden korjaaminen

Muotovirheiden korjaamisessa auttaa kone. Jos yrität käyttää ohjelmaa, jossa ei ole käskyä END, kone painaa paperillesi

NO END STATEMENT

ja lopettaa ohjelman käsittelyn. Osan mahdollisista virheistä kone löytää kääntäessään ohjelman omalle kielelleen, osan vasta suorittaessaan sitä.

Jos huomaat kirjoittaneesi jonkun rivin, esimerkiksi

```
10 ILPUT K
```

väärin, saat ohjelman kuntoon kirjoittamalla rivin uudelleen. Siis kirjoita

```
10 INPUT K
```

Tietokone ei välitä lauseiden kirjoitusjärjestyksestä, vaan suorittaa ohjelman rivinumeroiden tai käskyjen GOTO, IF-THEN osoittamassa järjestyksessä.

Jos ohjelmassa on useita samannumeroisia lauseita tietokone suorittaa niistä vain viimeisen. Lauseet joissa on vain rivinumero tietokone sivuuttaa. Lause voidaan siis mitätöidä kirjoittamalla tyhjä samannumeroinen lause. Esimerkiksi lause

```
10 LET A = 3
```

voidaan mitätöidä kirjoittamalla myöhemmin

```
10
```

Rivi voidaan tuhota välittömästi sen kirjoittamisen jälkeen painamalla näppäintä 'ESC', jos näppäintä "RETURN" ei ole ehditty painaa.

Merkkejä voidaan tuhota yksitellen painamalla ylätasossa olevaa merkkiä "←" eli samanaikaisesti näppäimiä "SHIFT" ja "O".

VIII Funktiot

BASIC-kieleen on valmiiksi ohjelmoitu helposti käytettävään muotoon sellaisia ohjelmia, joita käyttäjät usein tarvitsevat. Näitä osia nimitetään sisäisiksi funktioiksi.

Esimerkiksi lause

```
10 LETA = SQR (4)
```

antaa muuttujan A arvoksi luvun 4 neliöjuuren 2.



Luettelo funktioista

Funktio	Merkitys	Huomautuksia
SIN (X)	sin X	
COS (X)	cos X	argumentti radiaaneina
TAN (X)	tan X	
ATN (X)	arc tan X	tulos radiaaneina
EXP (X)	e^X	
LOG (X)	lnX	
ABS (X)	$\lvert x \rvert$	
SQR (X)	$\sqrt{x}$	
RND (X)	satunnaisluku väliltä (0,1)	
INT (X)	typistää x:n lähinnä pienemmäksi kokonaisluvuksi.	

Suluissa oleva argumentti voi olla lauseke, joka saattaa sisältää funktioita.

On olemassa käsky DEF, jolla ohjelmoija voi itse luoda tällaisia funktioita.

IX BASIC-käskyt

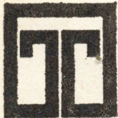
DATA luku 1, luku 2,.....

on käsky, jolla ohjelmoija sijoittaa käsiteltävää tietoa koneen muistiin otettavaksi käyttöön käskyllä READ. Näiden käskyjen toiminta on esitetty luvussa III olevassa esimerkissä.

DEF FN kirjain (muuttuja) = lauseke

Tällä käskyllä määritellään myöhemmin samassa ohjelmassa käytettävä funktio. Funktion nimi on kolmikirjaiminen ja kahden ensimmäisen kirjaimen tulee olla FN. Esimerkiksi

```
10 DEFFNA (X) = INT (X*100 + 0.5)/100
20 INPUT A,B
30 LET C = FNA (A*B)
40 PRINT C
```



tietotehdas oy

DIM taulukon nimi I (luku I, luku 2)),...

Tällä käskyllä varataan indeksoiduille muuttujille sopiva tila. Sitä on aina käytettävä kun indeksi on suurempi kuin 10.

END
ilmoittaa ohjelman käskyluettelon lopun.

FOR muuttuja = lauseke I TO lauseke 2 STEP lauseke 3

aloittaa silmukan, antaa muuttujalle alkuarvon (lauseke I), kasvattaa sitä jokaisella suoritus-kerralla, lauseke 3:n verran ja lopettaa silmukan läpikäymisen kun muuttujan arvo on suurempi kuin lauseke 2.

NEXT muuttuja
kertoo mihin käskyn FOR aloittama silmukka loppuu.

GOSUB rivinumero I
siirtää koneen suorittamaan rivinumero I:stä lähtien aliohjelmaa ja tallettaa seuraavan käskyn osoitteen paluuta varten.

RETURN
palauttaa koneen suorittamaan ohjelmaa sen GOSUB-käskyn jälkeisestä lauseesta, josta hyppy aliohjelmaan tapahtui.

GOTO rivinumero -I
siirtää koneen suorittamaan lausetta rivinumero-I:stä lähtien.

IF ehto-I THEN rivinumero-I

siirtää koneen suorittamaan lausetta rivinumero-I, jos ehto-I on tosi, mutta antaa koneen suorittaa lauseet järjestyksessä, jos ehto-I ei ole tosi.

INPUT muuttuja 1, muuttuja 2,.....

saa koneen painamaan listalle kysymysmerkin sekä odottamaan, kunnes käyttäjä on antanut arvot kaikille luettelossa mainituille muuttujille ja painanut näppäintä "RETURN".

LET muuttuja = lauseke
laittaa koneen laskemaan yhtäläisyysmerkin oikealla puolen olevan lausekkeen ja sijoittamaan tuloksen vasemmalla puolen olevan muuttujan arvoksi. Lauseke

```
10 LET A = A + 1
```

on sallittu ja tämän tyyppiset lauseet ovat jopa yleisiä

PRINT { lauseke 1 / "huomautus 1" } { ; / , } { lauseke 2 / huomautus 2 } ....

painaa joko lausekkeiden arvot tai huomautukset välilistalle. Jos välissä on pilkku, se painaa jäljessä seuraavan arvon tai huomautuksen uuden 15 sarakkeen lohkon alusta lähtien. Jos välissä on pilkku on tabulointi tiheämpää.

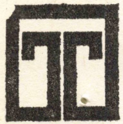
READ muuttuja 1, muuttuja 2, ...

lukee ensimmäisen käyttämättömän DATA-varastossa olevan luvun muuttuja 1:n arvoksi, toisen muuttuja 2:n arvoksi ja niin edelleen.

REM vain ohjelmalistaan tuleva kommentti on tarkoitettu ohjelman ymmärtämistä auttavien lauseiden säilyttämiseen.

RESTORE
antaa mahdollisuuden käyttää DATA-varasto uudelleen.

STOP
käskee koneen lopettaa ohjelman suorituksen.



tietotehdas oy

X Silmukat

Alla oleva ohjelma on päättymätön silmukka.

```
10 LETA = 1
20 GOTO  10
30 END
```

Rivillä kymmenen kone asettaa ykkösen A:n arvoksi ja seuraavalla rivillä on käsky siirtyä tekemään sama uudelleen. Tämä ohjelma on paitsi hyödytön myös vaarallinen, koska siinä ei ole yhtään tulostuskäskyä. Se ei kerro käyttäjälle ajon aikana mitään. Jos ohjelmaan jää tällainen silmukka, kone ehtii yleensä kuluttaa keskusyksikköaikaa kohtuuttomasti, ennen kuin ajo huomataan keskeyttää.

Jokaisessa silmukassa tulisi olla testikäsky, joka rajoittaa silmukan suorituskertojen määrää. Yleinen tapa pitää kirjaa siitä, kuinka monta kertaa kone on suorittanut silmukan, on laskurin sijoittaminen ohjelmakierroksen sisälle. Laskuri on muuttuja, jonka arvoa kone kasvattaa tavallisesti ykkösellä jokaisella kierroksella.

Alla on ohjelma, jossa kymmenen kertaa suoritettava silmukka. Ohjelmassa on laskuri, muuttuja X, jonka arvo on siihen mennessä suoritettujen kierrosten määrä. Tätä arvoa testataan rivillä 40 olevalla ehtolauseella, joka lopettaa silmukan suorittamisen kymmenennen kerran jälkeen.

```
10 LET X = 1
20 PRINT X, SQR (X)
30 LET X = X + 1
40 IF X > 10 THEN 60
50 GOTO 20
60 END
```



Silmukat

Alla oleva ohjelma on päättymätön silmukka.

```
10 LET A = 1
20 GOTO 10
30 END
```

Rivillä kymmenen kone asettaa ykkösen A:n arvoksi ja seuraavalla rivillä on käsky siirtyä tekemään sama uudelleen. Tämä ohjelma on paitsi hyödytön myös vaarallinen, koska siinä ei ole yhtään tulostuskäskyä. Se ei kerro käyttäjälle ajon aikana mitään, mos ohjelmaan jää tällainen silmukka, kone ehtii yleensä kuluttaa keskusyksikköaikaa kontuuttomasti ennen kuin ajo huomataan keskeyttää.

Jokaisessa silmukassa tulisi olla testikäsky, joka rajoittaa silmukan suorituskertojen määrää. Tietenkin tapa pitää kirjaa siitä, kuinka monta kertaa kone on suorittanut silmukan, on laskuri, sillä tiedämme ohjelmakerroksen sisällön. Laskuri on muuttuja, jonka arvoa kone kasvattaa tavallisesti ykkösellä jokaisella kierroksella.

Alla on ohjelma, jossa kymmenen kertaa suoritettava silmukka. Ohjelmassa on laskuri, muuttuja X, jonka arvo on siihen mennessä suoritettujen kierrosten määrä. Tätä arvoa testataan rivillä 40 olevalla ehtolauseella, joka lopettaa silmukan suorittamisen kymmenennen kerran jälkeen.

```
10 LET X = 1
20 PRINT X, SQR (X)
30 LET X = X + 1
40 IF X > 10 THEN 60
50 GOTO 20
60 END
```

Tällaiset ohjelmat ja ohjelman osat ovat niin yleisiä, että niiden tekemistä on helpotettu lisäämällä BASIC-kieleen erityinen kaksiosainen silmukkakäsky, FOR-NEXT. Ensimmäinen tämän käskyn osa kertoo, mistä silmukka alkaa, mikä on laskurin nimi ja mitä arvoja laskuri voi saada. Toinen osa kertoo, mihin silmukkaan kuuluvat käskyt loppuvat.

Edellinen esimerkki näyttää käskyä, FOR-NEXT, käyttäen tällaiselta:

```
10 FOR X = 1 TO 10
20 PRINT X, SQR(X)
30 NEXT X
40 END
```

Silmukoita voidaan kirjoittaa sisäkkäin. Esimerkiksi lista siitä, kuinka monta pulloa viidestä kymmeneen pullonpuhalluskonetta tekee yhdessä, kahdessa tai kolmessa vuorossa, kun yksi kone puhaltaa 123 pulloa tunnissa, voidaan kirjoittaa näin

```
10 FOR K = 5 TO L/
20 FOR T = 8 TO 24 STEP 8
30 PRINT K "KONETTA";T;"TUNNISSA"
40 PRINT "VALMISTAA"; K*T 123; "PULLOA"
50 NEXT T
60 NEXT K
70 END
```

Sen sijaan silmukoita ei saa lomittaa. Jos silmukka alkaa toisen sisältä, on sen viimeisenkin käskyn oltava ulomman silmukan sisällä.

Tällaiset ohjelmat ja ohjelman osat ovat niin yleisiä, että niiden tekemistä on helpotettu lisäämällä BASIC-kieleen erityinen kaksiosainen silmukkakäsky, FOR-NEXT. Ensimmäinen tämän käskyn osa kertoo, mistä silmukka alkaa, mikä on laskurin nimi ja mitä arvoja laskuri voi saada. Toinen osa kertoo, mihin silmukkaan kuuluvat käskyt loppuvat.

Edellinen esimerkki näyttää käskyn, FOR-NEXT, käyttäen tällaiselta:

```
10 FOR X = 1 TO 10
20 PRINT X, SQR(X)
30 NEXT X
40 END
```

Silmukoita voidaan kirjoittaa sisäkkäin. Esimerkiksi lista siitä, kuinka monta pulloa viidestä kymmeneen pullonpuhalluskonetta tekee yhdessä, kahdessa ja kolmessa vuorossa, kun yksi kone puhaltaa 125 pulloa tunnissa, voidaan kirjoittaa näin:

```
10 FOR K = 5 TO 10
20 FOR T = 8 TO 24 STEP 8
30 PRINT K "KONETTA";T;"TUNNISSA"
40 PRINT "VALMISTAA"; K*T*125; "PULLOA"
50 NEXT T
60 NEXT K
70 END
```

Sen sijaan silmukoita ei saa lomittaa. Jos silmukka alkaa toisen sisältä, on sen viimeisenkin käskyn oltava uloimman silmukan sisällä.


tietotehdas oy

XI Listat ja taulukot

Aina ei kannata antaa kaikille muuttujille omaa nimeä, vaan on helpompi käyttää yhteistä nimeä muuttujaryhmästä. Yksityisiin muuttujiin viitataan tällöin nimen perässä suluissa olevan järjestysluvun avulla.

Esimerkiksi A(3) kertoo ohjelmoijan viittaavan muuttujaryhmän A kolmanteen alkioon. Tällaista muuttujaryhmää kutsutaan listaksi tai vektoriksi.

Listan jäljessä sulkeissa voi olla myös muuttujan nimi tai lauseke. Tällöin sen arvo kertoo, monenteenko listan muuttujaan viitataan.

Esimerkiksi viiden myyntimiehen suhteellinen osuus koko myynnistä lasketaan kätevästi näin:

```
10 LET T = 0
20 FOR X = 1 TO 5
30 PRINT "MIKA ON";X;":N MIEHEN MYYNTI";
40 INPUT S(X)
50 LET T = T + S(X)
60 NEXT X
70 FOR X = 1 TO 5
80 PRINT X;":N MIEHEN OSUUS ON"; 100*S(X)/T
90 NEXT X
95 END
```

Muuttujaryhmät voidaan edelleen koota ryhmiksi, joita kutsutaan matriiseiksi tai taulukoiksi.

Taulukon nimen jäljessä on sulkeissa kaksi lukua, muuttujaa tai lauseketta. Jälkimmäinen niistä kertoo, mitä edellisen osoittaman taulukon listan alkiota halutaan käsitellä.



tietotehdas oy

Oletetaan, että yhtiössä on viisi myyntimiestä, joista jokainen myy kolmea tuotetta.

Ohjelma, joka laskee kunkin myyntimiehen osuuden kunkin tuotteen myynnistä, voidaan kirjoittaa esimerkiksi näin :

```
 20 PRINT "MITKA OVAT TUOTTEIDEN HINNAT";
 30 INPUT H(1), H(2), H(3)
 40 FOR A = 1 TO 3
 50 LET T = 0
 60 FOR B = 1 TO 5
 70 READ S(A,B)
 80 LET T = T + S(A,B)*H(A)
 90 NEXT B
100 PRINT "MYYNTIMIESTEN OSUUDET"
110 PRINT A;":N TAVARAN MYYNISTA"
120 PRINT "I";"II"; "III"; "IV"; "V"
130 PRINT 100*S(A,1)/T,100*S(A,2); 100*S(A,3)/T;
140 PRINT 100*S(A,4)/T; 100*S(A,5)/T
150 NEXT A
160 END
170 DATA 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
```

Jokaiselle ryhmämuuttujalle, listalle tai taulukolle on varattava ohjelman alussa tilaa käskyllä DIM, jos indeksin suurin arvo on suurempi kuin kymmenen. Näin on syytä menetellä tilan säästämiseksi, vaikka indeksi olisi pienempi.

Päinvastoin kuin useimmissa muissa kielissä on BASIC-listoissa ja taulukoissa myös nollas alkio, esimerkiksi A(0). Tämä on tärkeää muistaa, kun käytetään kokonaisten taulukoiden operointiin tarkoitettuja käskyjä, joita on yksitoista.



Matriisikäskyluettelo	Esimerkki
Kahden matriisin yhteenlasku	99 MAT C = A + B
Kahden matriisin vähentäminen toisistaan	99 MAT C = A - B
Matriisin kertominen toisella	99 MAT C = A*B
Matriisin kertominen reaaliluvulla	99 MAT B = (K)*A
Matriisin kääntäminen	99 MAT B = INV(A)
Matriisin transponointi	99 MAT B=TRN(A)
Matriisin alkioiden muuttaminen ykkösiksi	99 MAT A = CON(M,N)
Matriisin muuttaminen ykkösmatriisiksi	99 MAT A = IND (N,N)
Matriisin alkioiden muuttaminen nolliksi	99 MAT A = ZER (M,N)
Matriisin lukeminen	99 MAT READ A,B,C
Matriisin painaminen	99 MAT PRINT A (1,2),B;C

Jokaisen matriisikäskyillä käsiteltävän taulukon ulottuvuudet on määrättävä DIM-käskyllä. Koska taulukoissa on nollarivi ja nollasarake, varataan tila 3 x 4 matriisille A käskyllä DIM A(2,3).

Käskyillä MAT-CON, MAT-IDN, MAT-ZER ja MATREAD voidaan matriisin DIM-käskyssä määrättyjä ulottuvuuksia pienentää.

Esimerkiksi matriisinkääntöohjelmassa varataan DIM-käskyllä tarpeeksi tilaa. Sen jälkeen luetaan käännettävän matriisin todellinen dimensio, supistetaan matriisit oikeaan kokoon MAT-CON-käskyllä ennen käännöskäskyä MAT-INV.



tietotehdas oy


```
10 DIM A(30,30), B(30,30)
20 READ N
30 MAT A = CON (N-1, N-1)
40 MAT B = CON (N-1, N-1)
50 MAT READ A
60 MAT B = INV (A)
70 MAT PRINT B
80 DATA 3
90 DATA 1,2,3,4,5,6,7,8,9
95 END
```

Huomautus: Listojen ja taulukoiden nimet koostuvat yhdestä kirjaimesta.